Journal of Computer and System Sciences **59**, 70–83 (1999)
Article ID jcss.1999.1621, available online at http://www.idealibrary.com on IDEAL®

The Learnability of Unions of Two Rectangles in

Zhixiang Chen

Department of Computer Science, University of Texas—Pan American, 1201 West University Drive, Edinburg, Texas 78539
E-mail: chen@cs.panam.edu

and

Foued Ameur

Heinz Nixdorf Institute and Department of Mathematics and Computer Science, University of Paderborn, D-33095 Paderborn, Germany
E-mail: ameur@uni-paderborn.edu

Received March 6, 1995; revised December 10, 1998

We study the problem of properly learning unions of two axis-parallel rectangles over the domain $\{0, n-1\}^2$ in the on-line model with equivalence queries. When only $O(\log n)$ equivalence queries are allowed, this problem is one of the five interesting open problems proposed by W. Maass and G. Turán (*Mach. Learning* **14**, 1994, 251–269), regarding learning geometric concepts. In this paper, we design an efficient algorithm that properly learns unions of two rectangles over the domain $\{0, n-1\}^2$ using $O(\log^2 n)$ equivalence queries. © 1999 Academic Press

1. INTRODUCTION

We consider the model of on-line learning from examples introduced by Angluin [4] (see also [20, 22, 23]). In this model, the learning process may be viewed as a game between two players called *teacher* and *learner*. They use a set X, called the *domain of examples*, and a set of $\mathscr{C} \subseteq 2^X$, called the concept class. Before the game starts the teacher chooses an element $c_t \in \mathscr{C}$, called a *target concept*. The task of the learner is to identify c_t from examples. The game proceeds in iterations. During iteration j:

* The main result in this paper was presented in [11]. Constructions given in this paper are based on certain new design techniques and substantially simpler than those in [11]. The first author was supported by NSF grant CCR91-03055 when he was at Boston University.

0022-0000/99 $30.00

(i) the learner A proposes a *hypothesis* h_j^A from a hypothesis class $\mathscr{H} \subseteq 2^X$ and asks the teacher an *equivalence query* "$h_j^A \equiv c_t$?." The choice of h_j^A is determined by the current strategy of A.

(ii) if $h_j^A \equiv c_t$, then the teacher responds with "YES" and terminates the learning process. Otherwise he gives a *counterexample* (CE) $x \in X$ from the symmetric difference

$$h_j^A \,\Delta c_t = (c_t \backslash h_j^A) \cup (h_j^A \backslash c_t).$$

If a CE belongs to $c_t \backslash h_j^A$, then we call it a *positive* counterexample (PCE for short). The CE's belonging to $h_j^A \backslash c_t$ are called *negative* counterexamples (NCE for short).

The goal of the learner is to identify the target concept with a minimal number of equivalence queries. For the worst case analysis, we can imagine that the teacher and learner are adversaries and the teacher tries to make the task of the learner as hard as possible; i.e., he obliges the learner to make the maximal number of equivalence queries. This leads to the following:

(iii) the *learning complexity* of an algorithm A, denoted by $LC(A)$, is defined as

$$LC(A) = \max \left\{ i \in N \;\middle|\; \begin{array}{l} \text{there is } c_t \in \mathscr{C} \text{ and a learning process with} \\ \text{CE's } x_j \in h_j^A \,\Delta c_t \text{ such that} \\ h_j^A \not\equiv c_t \text{ for } j = 1, ..., i-1. \end{array} \right.$$

(iv) the *learning complexity* of a concept class C is defined by

$$LC(\mathscr{C}) = \min\{LC(A) \mid A \text{ is a learning algorithm for } \mathscr{C}\}.$$

At this stage, we want to mention that in the on-line model of Angluin [4] we distinguish between *proper* learning (the hypotheses proposed by the learner are from the target concept class, i.e., $\mathscr{H} = \mathscr{C}$) and *arbitrary* learning (the hypotheses of the learner are arbitrary concepts, i.e., $\mathscr{H} = 2^X$. In this paper we shall consider only proper learning algorithms.

We say that a learning algorithm for a concept class $\mathscr{C}$ is efficient if the learning complexity of the algorithm is polynomial in the logarithm of the size of the domain. The given definition of the learning complexity does not take into account the time spent by the learning algorithm A to compute its new hypothesis from the old hypotheses and the examples presented. There are cases for which the computation of such a hypothesis is not possible in polynomial time. The attention is focused only on the amount of interaction between the teacher and the learner, i.e., the number of CE's presented by the teacher. However, in this paper we are interested in learning algorithms that have run-time polynomial in d and $\log n$ as well.

One of the most important open problems in computational learning theory is that of efficient learnability of DNF formulas. Great efforts have been devoted to solve this problem in different models of learning. Because of the tight relation existing between the class of DNF formulas and the geometric classes studied in this paper we shall give a short overview on important results about learnability of DNF formulas.

Pitt and Valiant showed in [28] that for any constant $k \geqslant 2$, the class of k-term DNF formulas is not properly learnable in the *PAC* model (see [29] for definition) under the assumption that $RP \neq NP$. Their result implies that the class of k-term DNF formulas, for constant $k \geqslant 2$, is not properly learnable in the exact learning model using equivalence queries under the assumption that $P \neq NP$. Bshouty *et al.* showed in [10] that the class of $\sqrt{\log n}$-term DNF formulas is properly on-line learnable using equivalence and membership queries. It was shown in [26] that this positive result cannot be significantly improved in the exact model or the *PAC* model allowing membership queries, given certain standard theoretical complexity assumptions.

When the number of occurrences of each variable in a DNF formula is restricted, many positive and negative results have been obtained. Angluin *et al.* proved in [5] that the class of read-once Boolean formulas is properly learnable. In particular, this result implies that the class of read-once DNF formulas is properly learnable. Aizenstein *et al.* proved in [1] that the class of read-thrice DNF formulas is not properly learnable using equivalence and membership queries if $co\text{-}NP \neq NP$. On the other hand, it has been shown through the work in [18, 2, 27] that the class of read-twice DNF formulas is properly learnable using equivalence and membership queries. In [26] Pillaipakkamnat and Raghavan proved that the negative result in [1] still holds when one assumes $P \neq NP$, and they also establised many other negative results regarding proper learnability of subclasses of DNF formulas.

Although unions of rectangles are generalizations of DNF formulas, no significant progress has been made on the proper learnability of unions of rectangles. In [24] Maass and Turán proposed five interesting open problems regarding learning discritized geometric concepts. The first one is whether unions of two rectangles over the discretized plane $\{0, n-1\}^2$ is properly learnable using $O(\log n)$ equivalence queries.

In this paper, we shall study proper learnability of unions of two rectangles in the two-dimensional discretized space $\{0, ..., n-1\}^2$ with equivalence queres. We denote by N the set of all natural numbers. $\forall i, j \in N$, we use $[i, j]$ to denote the set $\{i, ..., j\}$ if $i \leqslant j$ or $\emptyset$ otherwise. We define the class of all discretized *axis-parallel rectangles* (or *rectangles* for short) over the domain $[0, n-1]^d$ as

$$BOX_n^d = \left\{ \prod_{i=1}^{d} [a_i, b_i] \,\middle|\, 0 \leqslant a_i, b_i \leqslant n-1, \forall i \in [1, d] \right\}.$$

The concept class of unions of two rectangles over the domain $[0, n-1]^2$ is denoted by

$$TWO_n^2 = \{ C_1 \cup C_2 \mid C_1, C_2 \in BOX_n^2 \}.$$

We organize this paper as follows. In Section 2, we survey previous research on learning unions of rectangles. In Section 3, we prove several technical results about the structures of unions of two rectangles over the domain $[0, n-1]^2$. In Section 4, we construct an algorithm that properly learns any union of two rectangles over the domain $[0, n-1]^2$ using $O(\log^2 n)$ equivalence queries. We list three open problems in Section 5.

2. PREVIOUS RESULTS

In the PAC model, Blumer *et al.* proved in [8] that for constant dimension d, the class of unions of nondiscretized rectangles over the d-dimensional Euclidean space is PAC learnable. Long and Warmuth proved in [21] that for constant k, the class of unions of k nondiscretized rectangles over arbitrary dimensional Euclidean space is learnable. For constant n, Jackson proved in [19] that any union of polynomially many discretized rectangles over the domain $[0, n-1]^d$ is strongly PAC learnable with respect to the uniform distribution and using membership queries as well.

For learning the concept class BOX_n^d the algorithm that issues the smallest rectangle consistent with all previous CE's is $2d$-space bounded and its efficiency has been proved in the PAC learning model. On the other hand, this strategy has a learning complexity $\Omega(dn)$ in the learning model of Angluin [4].

Maass and Turán [24] presented an algorithm that learns separately each of the 2^d corners of the target concept from BOX_n^d. Their algorithm has learning complexity $O(2^d \log n)$.

The best known on-line learning algorithm for BOX_n^d has been presented by Chen and Maass in [15, 16]. Their algorithm consists of $2d$ separate search strategies that determine the parameters $a_1, b_1, ..., a_d, b_d$ of the target concept $c_t = \prod_{j=1}^{d} [a_j, b_j]$. The learning complexity of their algorithm is $O(d^2 \log n)$.

In [6] Auer discussed the problem of learning the class of BOX_n^d in a noisy environment.[1] He showed that BOX_n^d is learnable if and only if the fraction of the noisy examples is less than $1/(d+1)$. For BOX_n^d he also presented a learning algorithm that requires $O[d^3 \log n/(1-r(2d+1))]$ equivalence queries, if the fraction of noise r is less than $1/(2d+1)$. Maass and Turán [24] also showed that even if the learner is allowed to propose arbitrary concepts as hypotheses, the learning complexity of BOX_n^d is $\Omega(d \log n)$. As shown by Auer and Long [7] this lower bound holds even if the membership queries are allowed. If we consider only proper learning, then this lower bound can be raised to $\Omega[(d^2/(\log d)) \log n]$ (see [6]). Ameur constructed in [3] a $2d$-space bounded algorithm that also properly learns BOX_n^d using $O(d^2 \log n)$ equivalence queries.

Maass and Warmuth developed in [25] a learning algorithm that matches the $\Omega(d \log n)$ lower bound. The hypotheses of their algorithms are represented by a "virtual threshold gate" of depth 1 that has $2dn$ boolean variables as input. It is still open whether one can close the "$(\log d)$-gap" between the upper and lower bounds in the model of proper learning. One should note that it follows from Angluin [4] that on-line learning with only equivalence queries implies PAC learning under any distribution.

[1] Environment is *noisy*, if some of the counterexamples are invalid or *noisy*, i.e., they belong to the target concept but are classified as negative or are outside the target concept but classified as positive.

When the learner is allowed to use both equivalence and membership queries, Chen and Homer [14] first proved that the class of unions of k rectangles over the domain $[0, n-1]^2$ is learnable with $O(k^3 \log n)$ queries. Later, Goldberg *et al.* [17] proved that for constant dimension d, the class of unions of rectangles over the domain $[0, n-1]^d$ is polynomial time learnable with equivalence and membership queries. They also proved that for constant k, but arbitrary dimension d, the class of unions of k rectangles is polynomial time learnable with equivalence and membership queries. Recently, it has been proved that for constant dimension d, the class of unions of rectangles over the domain $[0, n-1]^d$ is polynomial time learnable using only equivalence queries (see [9, 14, 25]).

3. STRUCTURAL PROPERTIES OF TWO_n^2

In this section we will show several structural properties about unions of two rectangles over the domain $[0, n-1]^2$. In the next section, we will use those properties to design an algorithm that properly learns TWO_n^2 using $O(\log^2 n)$ equivalence queries.

For any set $A \subseteq [0, n-1]^2$, we use $\Re(A)$ to denote the minimal rectangle in BOX_n^2 containing A.

Given $C \in TWO_n^2$, for any example $y \notin C$ and for any set of examples $S \subseteq C$, we say that (y, S) is a witness for C if and ony if $y \in \Re(S)$. It is easy to see that $C \notin BOX_n^2$ if and ony if there is a witness for it.

LEMMA 3.1. *Assume that (y, S) is a witness for $C \in TWO_n^2$. Let $y = (y_1, y_2)$ and $\Re(S) = [a_1, b_1] \times [a_2, b_2]$. Then, there are examples $u = (u_1, u_2)$, $v = (v_1, v_2) \in S$ such that either $u \in [a_1, y_1] \times [y_2, b_2]$ and $v \in [y_1, b_1] \times [a_2, b_2]$ (in this case, we call (y, u, v) a type-1 witness for C), or $u \in [a_1, y_1] \times [a_2, y_2]$ and $v \in [y_1, b_1] \times [y_2, b_2]$ (in this case, we call (y, u, v) a type-2 witness for C).*

The structures of type-1 and type-2 witnesses are illustrated in Fig. 1.

Proof. Because $\Re(S)$ is minimal, there is at least one example $u \in S$ at the upper boundary $[a_1, b_1] \times [b_2, b_2]$ of $\Re(S)$. Assume that $u \in [a_1, y_1] \times [b_2, b_2]$. Again, because $\Re(S)$ is minimal, there are examples $v' \in S$ and $v'' \in S$ at the bottom and

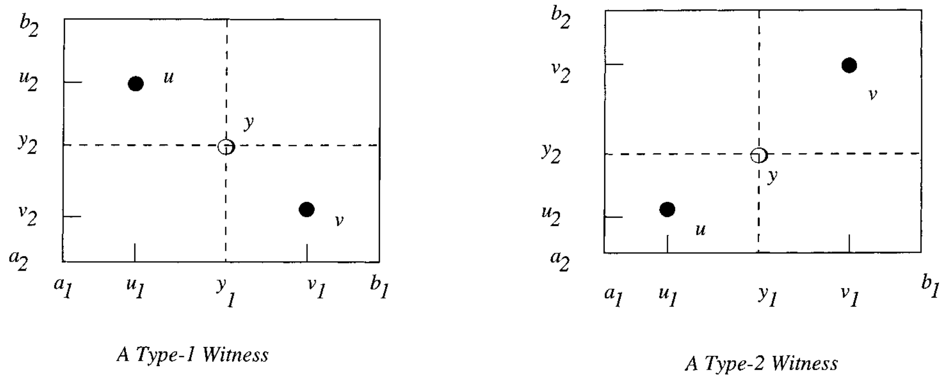

FIG. 1. Type-1 and type-2 witnesses.

right boundaries $[a_1, b_1] \times [a_2, a_2]$ and $[b_1, b_1] \times [a_2, b_2]$ of $\mathfrak{R}(S)$, respectively. If one of them, say, v' is in $[y_1, b_1] \times [a_2, y_2]$, then (y, u, v') is a type-1 witness for C. Otherwise, $v' \in [a_1, y_1] \times [a_2, y_2]$ and $v'' \in [y_1, b_1] \times [y_2, b_2]$; thus (y, v', v'') is a type-2 witness for C. Similarly, the lemma is also true when $u \in [y_1 + 1, b_1] \times [b_2, b_2]$. ∎

It has been shown in [15, 16] that there is an algorithm that properly learns BOX_n^d using $O(d^2 \log n)$ equivalence queries. Let LR denote a copy of the algorithm restricted over the domain $[0, n-1]^2$. Then, LR properly learns BOX_n^2 using at most $c \log n$ equivalence queries for a constant c.

LEMMA 3.2. *There is an algorithm that finds a witness for any target concept* $C \in TWO_n^2 \backslash BOX_n^2$ *using* $O(\log n)$ *equivalence queries.* (*Hence, by Lemma* 3.1, *the algorithm finds a type*-1 (*or a type*-2) *witness for* C.)

Proof. We employ algorithm LR to learn C. Since $C \notin BOX_n^2$ and the learner issues hypotheses in BOX_n^2 during the learning process of LR, the learner will not receive a "yes" from the teacher. Assume by contradiction that the learner has received $c \log n + 1$ CE's but has not found any witnesses. Let S be the set of all PCE's among the $c \log n + 1$ CE's. Thus, $\mathfrak{R}(S)$ is consistent with all those received CE's. Recall that $\mathfrak{R}(S) \in BOX_n^2$. Consider the learning process of LR on the target concept $\mathfrak{R}(S)$. Since algorithm LR is deterministic and is oblivious to the input target concept, the learning process of LR for $\mathfrak{R}(S)$ is the same as that for C for those $c \log n + 1$ CE's. Hence, the learner requires at least $c \log n + 1$ CE's to learn $\mathfrak{R}(S)$, a contradiction to the fact that $c \log n$ is the upper bound on the number of equivalence queries of LR. Therefore, the learner finds a witness (y, S) for C with at most $c \log n + 1$ CE's. ∎

Let $C \in TWO_n^2$. We say that C is *separable* if there are $A = \prod_{i=1}^2 [a_i, b_i]$ and $B = \prod_{i=1}^2 [e_i, f_i]$ such that, $C = A \cup B$ and $A \cap B = \emptyset$. It is easy to observe that $A \cap B = \emptyset$ if and only if one of the following conditions is true: (1) $b_1 < e_1$; (2) $f_1 < a_1$; (3) $b_2 < e_2$; and (4) $f_2 < a_2$. Thus, in other words, C is *separable* if and only if $C = A \cup B$ and one of the above four conditions is true.

Given $C = A \cup B = \prod_{i=1}^2 [a_i, b_i] \cup \prod_{i=1}^2 [e_i, f_i] \in TWO_n^2$, we say that C is an $S1$-shape union if $a_1 < e_1 \leqslant b_1 < f_1$ and $e_2 < a_2 \leqslant f_2 < b_2$. We say that C is an $S2$-shape union if it can be obtained by rotating an $S1$-shape union by 90°.

We say that C is an X-shape union, if $e_1 < a_1 \leqslant b_1 < f_1$ and $a_2 < e_2 \leqslant f_2 < b_2$.

It is easy to see that $S1$-shape, $S2$-shape, and X-shape unions are not separable. Examples of $S1$-shape, $S2$-shape, and X-shape unions are given in Fig. 2.

LEMMA 3.3. *For any* $C \in TWO_n^2 \backslash BOX_n^2$, *if it is not separable, then it is an* $S1$-*shape union, an* $S2$-*shape union, or an* X-*shape union.*

Proof. Let M be the minimal rectangle containing C. Because C is not in BOX_n^2 and not separable, M has four distinct corner points. Note that for a pair of rectangles which overlapped and formed an "L" (or a "T"), they could alternatively be expressed using a pair of nonoverlapping rectangles (hence, their union is separable).

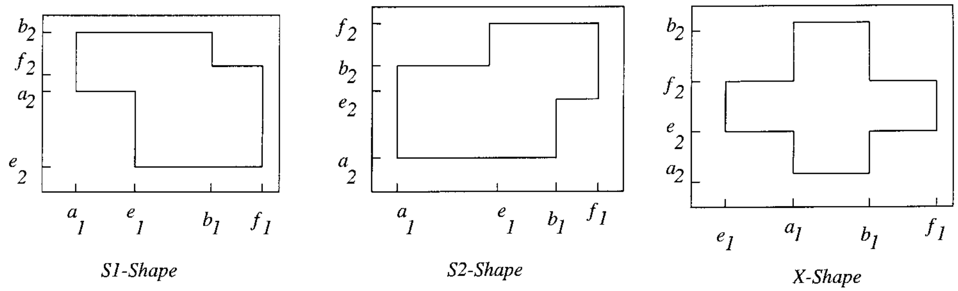

FIG. 2. $S1$-shape, $S2$-shape, and X-shape unions.

Let $C = A \cup B$. If either A or B contains two diagonal corner points of M, then $C = M$, a contradiction to $C \notin BOX_n^2$. Thus, neither A nor B contains two diagonal corner points of M. This implies that each of A and B may contain no corner points, one corner pointer, or adjacent corner points of M.

If A contains no corner points, then the only possibility to arrange B such that $A \cup B$ is not separable is that B contains no corner points and, A and B form an X-shape union touching all four boundaries of M.

If A contains one corner point, say the bottom left corner, then the only possibility to arrange B such that $A \cup B$ is not separable is that B contains the upper right corner only and, A and B overlap. Thus, A and B form a $S2$-shape. Similarly, if A contains the bottom right corner, then B contains the upper left corner, thus they form an $S1$-shape. With the same analysis, if A contains one of the two corners, then A and B form an $S1$-shape or an $S2$-shape.

If A contains two adjacent corner points, say, the two bottom corners, then no matter how to arrange B, their union is either a "T" or a "L" that is separable. This implies that A cannot contain two adjacent corner points.

The same analysis can be done for different cases of B. Putting the above together, C either contains no corner points of M or it contains two diagonal corner points. In the first case, C is an X-shape. In the latter case, C is either an $S1$-shape or an $S2$-shape. ▮

4. LEARNING TWO_n^2 USING EQUIVALENCE QUERIES

Maass and Turán [24] proposed five open problems regarding on-line learning geometric concepts. The first problem is whether the class of unions of two discretized axis-parallel rectangles over the domain $[0, n-1]^2$ is properly learnable using $O(\log n)$ equivalence queries. In this section, we provide a partial solution to the open problem by showing that the class of unions of two discretized axis-parallel rectangles over the domain $[0, n-1]^2$ is properly learnable using $O(\log^2 n)$ equivalence queries. The proof below is substantially different from the earlier one given in [11]. The proof in [11] is very complicated because it analyzes all possible cases and provides a particular solution for each of those cases.

LEMMA 4.1. *One can properly learn any separable target concept $C \in TWO_n^2$ using $O(\log^2 n)$ equivalence queries.*

Proof. Given a separable concept $C = A \cup B = \prod_{i=1}^{2} [a_i, b_i] \cup \prod_{i=1}^{2} [e_i, f_i]$, we know that one of the following conditions is true: (1) $b_1 < e_1$; (2) $f_1 < a_1$; (3) $b_2 < e_2$; and (4) $f_2 < a_2$. However, we do not know which one is true. We design a learning algorithm which will try each of the four conditions. Here, we only consider how the algorithm works under the condition $b_1 < e_1$. One possible case of the condition is illustrated in Fig. 3. The other three conditions can be coped with in a similar manner.

For any witness (y, S) for C, let $r(S) = (r_1, r_2)$ and $l(S) = (l_1, l_2)$ be two examples in S such that $\forall x = (x_1, x_2) \in S$, $l_1 \leqslant x_1 \leqslant r_1$. In other words, $r(S)$ is an example in S with the largest first coordinate, and $l(S)$ is an example in S with the smallest first coordinate. If $l(S) \in B$, then $S \subseteq B$ since $b_1 < e_1 \leqslant l(S)$. This implies $y \in \Re(S) \subseteq B$. Hence, $y \in C$, a contradiction of the fact that $y \notin C$. Thus, $l(S) \in A$. Similarly, $r(S) \in B$. Now, we can learn C as follows.

Let LA and LB be two copies of algorithm LR. The global algorithm uses LA and LB to learn A and B at stages. At each stage, when LA and LB issue respectively two hypotheses $H(A)$ and $H(B)$, the global algorithm issues a new hypothesis $H(A) \cup H(B)$. We use W to collect counterexamples that have been assigned to LA by the global algorithm since the last initiation of LA. We describe the learning algorithm below.

Initially, set $H(A) = H(B) = \varnothing$, *and set* $W = \varnothing$.
Repeat the following process:

> *Ask an equivalence query for* $H(A) \cup H(B)$. *The global algorithm stops if it receives "YES." If it receives a CE* x, *then it adds* x *to* W.
> *The global algorithm decides, among all CE's in* W, *whether there is a witness* (y, S) *for the target concept. If so, it gives* $r(S)$ *to LB to produce a new hypothesis and resets* $H(A) = \varnothing$ *and* $W = \varnothing$, *and thus it starts a new initiation of LA.*
> *If there is no witnesses, then if the received counterexample* x *is a PCE, then the global algorithm gives it only to LA to produce a new hypothesis and, lets*

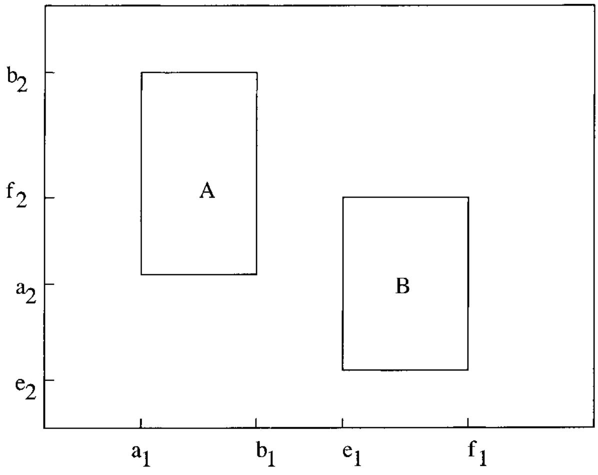

FIG. 3. A separable union with $b_1 < e_1$.

LB do nothing but issue the previous hypothesis; otherwise the global algorithm gives it to both LA and LB to produce two new hypotheses, respectively.

We now analyze the learning complexity of the above process. When the global algorithm finds a witness (y, S), then by the above analysis, $r(S) \in B$. Since $r(S)$ is a PCE to the union of LA and LB's hypotheses, it is not in LB's hypothesis. Since it is in B, it is a PCE for LB (learning B). So, LB always receives PCE's in B. Hence, LB learns B using $O(\log n)$ equivalence queries, since it is a copy of algorithm LR for learning BOX_n^2 using $O(\log n)$ equivalence queries. By Lemma 3.2, the global algorithm needs $O(\log n)$ equivalence queries to find a witness. Hence, the global algorithm needs $O(\log^2 n)$ equivalence queries to learn B. After that, all the PCE's received by the global algorithm are in A. Thus, LA can learn A using $O(\log n)$ additional equivalence queries, because LA is also a copy of algorithm LR for learning BOX_n^2 using $O(\log n)$ equivalence queries. Therefore, the global algorithm needs $O(\log^2 n)$ equivalence queries in total to learn A and B. ▮

LEMMA 4.2. *One can properly learn any $S1$-shape union in TWO_n^2 with $O(\log^2 n)$ equivalence queries. Similarly, one can properly learn any $S2$-shape union in TWO_n^2 with $O(\log^2 n)$ equivalence queries.*

Proof. We only consider $S1$-shape unions. Given any target concept $C = A \cup B = \prod_{i=1}^2 [a_i, b_i] \cup \prod_{i=1}^2 [e_i, f_i]$. By the definition of $S1$-shape unions, we have $a_1 < e_1 \leqslant b_1 < f_1$ and $e_2 < a_2 \leqslant f_2 < b_2$ (see Fig. 2). It is easy to see that there are type-1 witness for C, but there are no type-2 witnesses for it. For any type-1 witness (y, u, v), one can verify from the definition that $u \in A$ and $v \in B$.

In a proof similar to the one we did for Lemma 4.1, the global algorithm employs two copies LA and LB of algorithm LR to learn A and B, respectively. The only exception is that, when one obtains a witness (x, S), by Lemma 3.1, the global algorithm can find a type-1 witness (y, u, v) among the examples in $S \cup \{x\}$. It then gives v to LB to produce a new hypothesis, resets the hypothesis of LA to empty, and starts a new initiation of LA. Analogously, the global algorithm properly learns C using $O(\log^2 n)$ equivalence queries. ▮

LEMMA 4.3. *One can properly learn any X-shape union in TWO_n^2 with $O(\log^2 n)$ equivalence queries.*

Proof. Given any X-shape target concept $C = A \cup B = \prod_{i=1}^2 [a_i, b_i] \cup \prod_{i=1}^2 [e_i, f_i]$, we have $e_1 < a_1 \leqslant b_1 \leqslant f_1$ and $a_2 < e_2 \leqslant f_2 < b_2$. It is easy to see that there are type-1 and type-2 witnesses for C.

Given any type-1 witness (y, u, v), then either $y \in [b_1, f_1] \times [f_2, b_2]$ or $y \in [e_1, a_1] \times [a_2, e_2]$. Those two cases are illustrated in Fig. 4. When $y \in [b_1, f_1] \times [f_2, b_2]$, we can easily verify the following.

PROPERTY 4.4.

1. $u \in A$ and $v \in B$.

2. For any type-1 witness (y', u', v'), if $y'_1 < u_1$, then $u' \in B$ and $v' \in A$; otherwise $u' \in A$ and $v' \in B$. Here, $u = (u_1, u_2)$, $y' = (y'_1, y'_2)$.

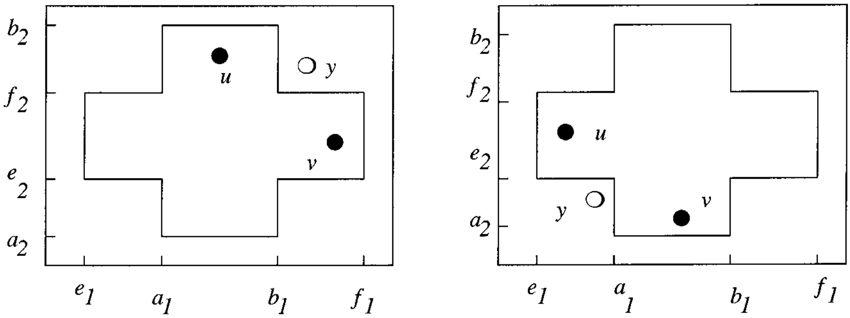

FIG. 4. Two possible structures of a type-1 witness for an X-shape union.

3. For any type-2 witness (y'', u'', v''), if $y''_1 < u_1$ then $u'' \in B$ and $v'' \in A$; otherwise $u'' \in A$ and $v'' \in B$. Here, $u = (u_1, u_2)$, $y'' = (y''_1, y''_2)$.

When $y \in [e_1, a_1] \times [a_2, e_2]$, we can also give similar properties like those in Property 4.4 to assign for any type-1 (or type-2) witness (y', u', v'), u' and v' to A and B correctly.

Symmetrically, given any type-2 witness (y, u, v), then either $y \in [e_1, a_1] \times [f_2, b_2]$ or $y \in [b_1, f_1] \times [a_2, e_2]$. In any of the two cases, one can assign, for any type-1 (or type-2) witness (y', u', v'), u' and v' to A and B corretly.

We now consider how to learn C. The learning process is divided into the following four parts. The control flow of the global algorithm is illustrated in Fig. 5.

Part 1: *Finding the first witness.* In the same way as we did in the proof of Lemma 4.1, the global algorithm employs two copies LA and LB of algorithm LR to learn A and B, respectively. However, when the global algorithm finds the first witness (x, S), it stops. Using Lemma 3.1, it then finds the first type witness (y, u, v), which is either type-1 or type-2, among the examples in $S \cup \{x\}$. Remember that the witness (y, u, v) will be kept by the global algorithm and will be used in Part 3 to assign CE's for LB to learn B.

Part 2: *Deciding whether the first type witness* (y, u, v) *is type*-1 *or type*-2. The global algorithm decides whether (y, u, v) is type-1 or type-2 according to the definition given in Lemma 3.1. This decision is deterministic and rather easy to be performed.

Part 3: *Trying the two possible locations for* y. If (y, u, v) is a type-1 witness, then $y \in [b_1, f_1] \times [f_2, b_2]$ or $y \in [e_1, a_1] \times [a_2, e_2]$. Unfortunately, the global algorithm does not know which of the two conditions is true. Similarly, if (y, u, v) is a type-2 witness, then $y \in [e_1, a_1] \times [f_2, b_2]$ or $y \in [b_1, f_1] \times [a_2, e_2]$. Unfortunately, the global algorithm does not know which of the two conditions is true, either. Our strategy is to allow the global algorithm to try each of the two conditions. More precisely, our strategy is as follows:

If (y, u, v) is a type-1 witness, then the global algorithm first guesses that $y \in [b_1, f_1] \times [f_2, b_2]$, and goes to Part 4 to continue to learn. If it learns the target concept C in Part 4, then it stops. If it does not learn the target concept in Part 4,

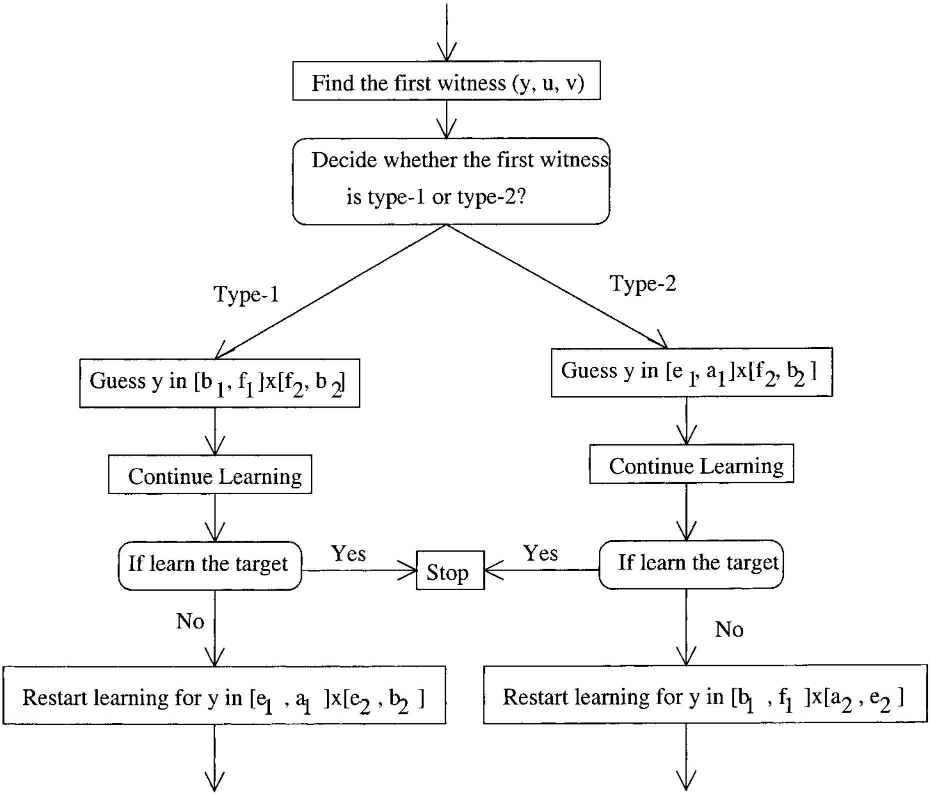

FIG. 5. The control flow for learning an X-shape union.

then it knows that y must be in $[e_1, a_1] \times [a_2, e_2]$. Hence, it uses the new condition $y \in [e_1, a_1] \times [a_2, e_2]$ to do Part 4 one more time.

Similarly, if (y, u, v) is a type-2 witness, then the global algorithm first guesses that $y \in [e_1, a_1] \times [f_2, b_2]$, and goes to Part 4 to continue to learn. If it learns the target concept C in Part 4, then it stops. If it does not learn the target concept in Part 4, then it knows that y must be in $[b_1, f_1] \times [a_2, e_2]$. Hence, it uses the new condition $y \in [b_1, f_1] \times [a_2, e_2]$ to do Part 4 one more time.

Part 4: *Using the first witness* (y, u, v) *and the location of* y *to learn the target concept* C. In the same way as in the proof of Lemma 4.1, the global algorithm employs two copies LA and LB of algorithm LR to learn A and B, respectively. During the learning process, whenever the global algorithm finds a new (type-1 or type-2) witness (y', u', v'), it will use the first witness (y, u, v) and the location of y, as well as Property 4.4, to determine which one of u' and v' belongs to B and thus, to assign it to the learning algorithm LB accordingly. Moreover, we only allow the global algorithm to continue learning for at most $t \log^2 n$ queries, where the constant t will be determined in the following paragraphs.

Now, assume that (y, u, v) is a type-1 witness and $y \in [b_1, f_1] \times [f_2, b_2]$. By Property 4.4, the global algorithm assigns v to LB to produce a new hypothesis and resets the hypothesis of LA to empty. After that, the global algorithm continues learning as it did in the proof of Lemma 4.1. Whenever it receives a new witness

(x', S'), by Lemma 4.1 it finds also a new type-1 (or type-2) witness (y', u', v'). Then, by Property 4.4, it assigns one of u' and v' to the learning algorithm LB. It then lets LB produce a new hypothesis and accordingly resets the hypothesis of LA to empty. With an analysis similar to the proof of Lemma 4.1, the global algorithm properly learns C using $O(\log^2 n)$ equivalence queries.

If (y, u, v) is a type-1 witness and $y \in [e_1, a_1] \times [e_2, a_2]$, with a similar analysis, the global algorithm can also learn C using $O(\log^2 n)$ queries. In the same way, we can show that the global algorithm learns C using $O(\log^2 n)$ equivalence queries, if (y, u, v) is a type-2 witness and $y \in [e_1, a_1] \times [f_2, b_2]$, or if (y, u, v) is a type-2 witness and $y \in [b_1, f_1] \times [a_2, e_2]$.

Choose a constant t such $t \log^2 n$ is the upper bound on the number of queries required by the global algorithm in each of the above four cases, then t is the constant needed in Part 4. ∎

THEOREM 4.5. *There is an algorithm that properly learns TWO_n^2 using $O(\log^2 n)$ equivalence queries.*

Proof. Let L_1, L_2, and L_3 be the algorithms constructed for Lemmas 4.1, 4.2, and 4.3, respectively. Fix a constant c such that $c \log^2 n$ is a common upper bound on the number of equivalence queries of L_1, L_2, and L_3. For any target concept $C \in TWO_n^2$, the global algorithm first employs L_1 to learn it for at most $c \log^2 n$ equivalence queries. If L_1 learns it, then the global algorithm stops. Otherwise, by Lemma 4.1, C is not separable. Thus, by Lemma 3.3, C is an $S1$-shape (or $S2$-shape, or X-shape) union. The global algorithm then employs L_2 to continue learning for at most $c \log^2 n$ equivalence queries. If L_2 learns it then the global algorithm stops. Otherwise, by Lemma 4.2, it is an X-shape union. Hence, by Lemma 4.3, the global algorithm can finally learn it by employing L_3 for at most $c \log^2 n$ queries. ∎

5. OPEN PROBLEMS

In [12], an efficient algorithm was constructed to properly learn unions of two rectangles over the domain $[0, n-1]^d$ with at most two equivalence queries and at most $(11d+2) \log n + d + 3$ membership queries. The proofs in [12] are based on case analysis and very complicated and tedious. We do not know whether one can find simpler constructions and proofs for the results obtained in [12].

Can one design an efficient algorithm that properly learns unions of k axis-parallel rectangles over the domain $[0, n-1]^d$ with equivalence and membership queries for any nonconstant k? It seems that this problem is not easy even if d is fixed.

Is $\Omega(\log^2 n)$ the lower bound on the number of equivalence queries for the proper learning of unions of two axis-parallel rectangles over the domain $[0, n-1]^2$?

ACKNOWLEDGMENTS

The authors are very grateful to the referee for reading this paper and checking the proofs, and for his (or her) suggestions about revising this paper.

REFERENCES

1. H. Aizenstein, L. Hellerstein, and L. Pitt, Read-thrice DNF is hard to learn with membership and equivalence queries, *in* "Proc. of the 33rd Annual IEEE Symposium on the Foundations of Computer Science, 1992," pp. 523–532.
2. H. Aizenstein and L. Pitt, Exact learning of read-twice DNF formulas, *in* "Proc. of the 33rd Annual IEEE Symposium on the Foundations of Computer Science, 1991," pp. 170–179.
3. F. Ameur, A space-bounded learning algorithm for axis-parallel rectangles, *in* "Proc. of the 2nd European Conference on Computational Learning Theory," Lecture Notes in Artificial Intelligence, Vol. 904, pp. 313–321, Springer-Verlag, New York/Berlin, 1995.
4. D. Angluin, Queries and concept learning, *Mach. Learning* **2** (1988), 319–342.
5. D. Angluin, L. Hellerstein, and M. Karpinski, Learning read-once formulas with queries, *J. Assoc. Comput. Mach.* **40** (1993), 185–210.
6. P. Auer, On-line learning of rectangles in noisy environment, *in* "Proc. of the 6th Annual ACM Workshop on Computational Learning Theory, 1993," pp. 253–261.
7. P. Auer and P. Long, Simulating access to hidden information while learning, *in* "Proc. of the 26th Annual ACM Symposium on the Theory of Computation, 1994," pp. 263–272.
8. A. Blumer, A. Ehrenfeucht, D. David, and M. Warmuth, Learnability and the Vapnik-Chervonenkis dimension, *J. Assoc. Comput. Mach.* **4** (1989), 929–965.
9. N. Bshouty, Z. Chen, and S. Homer, On learning discretized geometric concepts, *in* "Proc. of the 35th Annual IEEE Symposium on Foundations of Computer Science, 1994," pp. 54–63.
10. N. Bshouty, S. Goldman, T. Hancock, and S. Matar, Asking questions to minimize errors, *in* "Proc. of the 6th Annual ACM Conference on Computational Learning Theory, 1993," pp. 41–50.
11. Z. Chen, Learning unions of two rectangles in the plane with equivalence queries, *in* "Proc. of the 6th Annual ACM Conference on Computational Learning Theory, 1993," pp. 243–253.
12. Z. Chen, An optimal algorithm for proper learning of unions of two rectangles with queries, *in* "Proc. of the First International Computing and Combinatorics Conference, Lecture Notes in Computer Science," Vol. 959, pp. 334–343, Springer-Verlag, New York/Berlin, 1995.
13. Z. Chen and S. Homer, "Learning Unions of Rectangles with Queries," Technical Report BUCS-93-10, Dept. of Computer Science, Boston University, 1993.
14. Z. Chen and S. Homer, The bounded injury priority method and the learnability of unions of rectangles, *Annals of Pure and Applied Logic* **77** (1996), 143–168.
15. Z. Chen and W. Maass, On-line learning of rectangles, *in* "Proc. of the 5th ACM Workshop on Computational Learning Theory, 1992," pp. 16–28.
16. Z. Chen and W. Maass, On-line learning of rectangles and unions of rectangles, *Mach. Learning* **17** (1994), 201–223.
17. P. Goldberg, S. Goldman, and D. Mathias, Learning unions of rectangles with membership and equivalence queries, *in* "Proc. of the 7th Annual ACM Conference on Computational Learning Theory, 1994," pp. 198–207.
18. T. Hancock, Learning 2μ DNF formulas and $k\mu$ decision trees, *in* "Proc. of the 4th Annual ACM Conference on Computational Learning Theory, 1991," pp. 199–209.
19. J. Jackson, An efficient membership-query algorithm for learning DNF with respect to the uniform distribution, *in* "Proc. of the 35th Annual IEEE Symposium on Foundations of Computer Science, 1994," pp. 42–53.
20. N. Littlestone, Learning quickly when irrelevant attributes abound: a new linear threshold algorithm, *Mach. Learning* **2** (1987), 285–318.
21. P. Long and M. Warmuth, Composite geometric concepts and polynomial predictability, *in* "Proc. of the 3th Annual ACM Workshop on Computational Learning Theory, 1991," pp. 273–287.
22. W. Maass and G. Turán, On the complexity of learning from counterexamples, *in* "Proc. of the 30th Annual IEEE Symposium on Foundations of Computer Science, 1989," pp. 262–267.

23. W. Maass and G. Turán, On the complexity of learning from counterexamples and membership queries, *in* "Proc. of the 31th Annual Symposium on Foundations of Computer Science, 1990," pp. 203–210.

24. W. Maass and G. Turán, Algorithms and lower bounds for on-line learning of geometric concepts, *Mach. Learning* **14** (1994), 251–269.

25. W. Maass and M. Warmuth, "Efficient Learning with Virtual Threshold Gates," Technical Report 395 of the Institutes for Information Processing Graz, 1994.

26. K. Pillaipakkamnatt and V. Raghavan, On the limits of proper learnability of subclasses of DNF formulas, *in* "Proc. of the 7th Annual Conference on Computational Learning Theory, 1994," pp. 118–129.

27. K Pillaipakkamnat and V. Raghavan, "Read-twice DNF Formulas are Properly Learnable," Technical Report TR-93-58, Department of Computer Science, Vanderbilt University, 1993.

28. L. Pitt and L. G. Valiant, Computational limitations on learning from examples, *J. Assoc. Comput. Mach.* **35** (1988), 965–984.

29. L. Valiant, A theory of the learnable, *Comm. Assoc. Comput. Mach.* **27** (1984), 1134–1142.